République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Licence en Informatique

Thème

Gestion d'une bibliothèque avec Swing et Hibernate

Réalisé par :

- BENDAHMANE Zahira
- BELGHORZI Amel

Présenté le 09 Juin 2014 devant la commission d'examination composée de MM.

- ***Mr BADR Benmammar B.*** *(Encadreur)*
- ***Mr Bendaoud F.*** *(Co-Encadreur)*
- ***Matallah H.*** *(Examinateur)*
- ***Merzoug M.*** *(Examinateur)*

Année universitaire: 2013-2014

Remerciement

Nous tenons tout d'abord à remercier le bon ***Dieu*** qui nous a donné la santé et le courage, ainsi que l'audace pour dépasser toutes les difficultés et d'accomplir ce travail.

Nous tenons à exprimer notre profonde gratitude et notre respectueuse reconnaissance à notre encadreur ***Mr***. ***BENMAMMAR Badr*** qui a bien voulu nous encadrer. Nous le remercie vivement pour son soutien et ses conseils précieux et leur assistant ***Mr***. ***BENDAOUD Faycal***.

Nos remerciements les plus sincères à tous nos enseignants, ***Mr KADRI Benamar***, nos amis, tous le personnel de bibliothèque et tous ceux qui nous ont prêté mains fortes pour la réalisation de ce projet.

Nous remercions également les membres de jury qui nous font honneur en acceptant d'examiner et de juger notre travail.

Merci…

Dédicace

A l'aide de DIEU tout puissant, qui trace le chemin de ma vie, donc Je dédie ce travail :

A mes chers parents jamais je ne saurais m'exprimé quant aux sacrifices et aux dévouements que vous consacrés à mon éducation et mes études. Les mots expressifs soient-ils restent faibles pour énoncer ma gratitude hautement profonde.

Tous les mots ne sauraient exprimer la gratitude, l'amour, le respect, la reconnaissance

A mes frère Ali, Mohamed et mes sœurs Fathia, Malika

Et à toute la famille BELGHORZI sans Exception.

Un grand merci à celle qui m'a aidé à faire ce travail ma très chère amie Zahira et à toute sa famille.

A tous mes amis.

Melle Belghorzi Amel

Dédicace

Je dédie, avec respect ce modeste travail à tous ceux qui m'ont inspiré aimé et aidé de proche ou de loin :

Mes très chers parents

Je sais que vous étiez toujours fière de moi et j'espère que vous le serez plus aujourd'hui.

Que Dieu vous garde et vous alloue bonne santé, bonheur, et longue vie.

Mes très chers frères

Mohamed et Salem

Ma très chère sœur

Chahra et son mari Kada, leur fille Malek et leur garçon Mohamed Akram.

Un grand merci à celle qui m'a aidé à faire ce travail ma très chère amie Amel et à toute sa famille.

Mes très chères amies : Fatima, Nabila, Imène, Asma, Djahida, Asma, Karima, Oussama, Ismail

Tous mes collègues d'études.

Melle Bendahmane Zahira

Table des matières

SOMMAIRE

Table des matières

Table des matières

La liste des figures

INTRODUCTION GÉNÉRALE

INTRODUCTION GENERALE

L'informatique est sans doute la révolution la plus importante et la plus innovante qui a marqué la vie de l'humanité moderne. En effet, les logiciels informatiques proposent maintenant des solutions à tous les problèmes de la vie, aussi bien dans des domaines professionnels que pour des applications personnelles. De même pour les méthodes de conception et de développement qui ont vu l'avènement d'autant de technologies qui facilitent leur mise en place et leurs donnent des possibilités et des fonctionnalités de plus en plus étendues. [1]

Les bibliothèques doivent aujourd'hui relever les nouveaux défis posés par l'univers de l'information. Elles doivent notamment permettre aux utilisateurs d'accéder rapidement et simplement à une information pertinente, tout en accroissant l'efficacité opérationnelle de leurs collaborateurs. [2]

Le travail réalisé dans le cadre de notre projet vise à développer une application qui sert à faire la gestion d'une bibliothèque avec **Swing** et **Hibernate** qui permet aux différents services d'emprunter, chercher, ajouter et modifier d'un livre.

Notre mémoire est structuré en deux chapitres comme suit :

- **CHAPITRE 1 :**

C'est la partie théorique, il est divisé en deux parties :

La première partie débute par une vision sur la programmation réseaux, le modèle diffusion, le modèle P2P et le modèle client-serveur. La deuxième partie nous allons présenter les différents outils que nous allons utiliser comme **Java**, **NetBeans**, **EasyPHP**, **MySQL** et enfin le Framework « **Hibernate** » qui apporte des méthodes objet de haut niveau aussi évite les requêtes SQL.

- **CHAPITRE 2 :**

C'est la partie pratique qui contient tous les étapes nécessaires pour la réalisation de notre application en détail avec des captures écrans.

CHAPITRE I : PROGRAMMATION RÉSEAU ET OUTILS DE TRAVAIL

I. programmation réseau

I.1Introduction

Un réseau est un ensemble de matériels et de logiciels permettant à des équipements de communiquer entre eux. L'objectif d'un réseau est le partage des ressources matérielles (disques durs, imprimantes) et des ressources logicielles (fichiers, applications). Les réseaux regroupent un ensemble hétérogène d'architectures, du filaire au sans-fil, du LAN au WAN. [3]

Un système réparti est organisé comme un ensemble de processus qui s'exécutent sur des sites reliés par un réseau de communication et qui communique par envoi de messages. Sur chaque site, il est possible de définir un état local, qui est modifié par l'exécution des processus du site. [4]

On va présenter les trois types de modèles d'architectures suivantes : le modèle diffusion, le modèle P2P et le modèle client-serveur.

I.2 Modèle de diffusion

I.2.1 Diffusion de messages (broadcast)

On a deux rôles distincts :

- Emetteur : envoie des messages (ou événements) à destination de tous les récepteurs. Possibilité de préciser un sous-ensemble de récepteurs (multicast).
- Récepteurs : reçoivent les messages envoyés.

Emetteur envoie un message et le middleware s'occupe de transmettre ce message à chaque récepteur.

I.2.2Les deux modes de réception

Dans le modèle diffusion on a deux modes de réception :

- Le récepteur va vérifier lui-même qu'il a reçu un message (Boite aux lettres).
- Le récepteur est prévenu que le message est disponible et il lui est transmis (Le facteur sonne à la porte pour remettre en main propre le courrier).

I.2.3 Particularités du modèle

- Dépendance plus faible entre les participants (Pas besoin pour l'émetteur d'être directement connecté aux récepteurs ni même de savoir combien ils sont).
- Interaction de type « 1vers N ». [5]

I.3 Modèle P2P

I.3.1 Définition

Peer to Peer est un type de connexion réseau par laquelle deux machines communiquent d'égal à égal, à l'opposé des relations maître esclave. Ce type de connexion permet à des millions d'internautes affiliés à un réseau de partager leurs fichiers stockés sur le disque dur de leur machine. [6]

I.3.2 Modes d'architecture P2P

Il existe deux grandes variantes de l'architecture peer-to-peer :

- Architecture hybride (assistée par un serveur) : deux ordinateurs communiquent entre eux après la localisation du fichier recherché. La localisation de la ressource se fait via un serveur, connu par une communauté, contenant un annuaire commun où les utilisateurs s'enregistrent et déclarent les ressources à partager.
- Architecture native : chaque ordinateur se prête au rôle de client/serveur/moteur de recherche. Avec ce mode natif ou pur, on parle de communication entre ordinateurs sans l'intermédiaire d'un serveur, les usagers connectés peuvent échanger leurs fichiers sans passer par un serveur central à condition d'utiliser un même logiciel peer-to-peer. [7]

I.3.3 L'avenir du P2P

En informatique, difficile de prédire l'avenir, mais on peut voir quelques tendances:

- Les P2P payants : Actuellement, le contenu de tous les réseaux P2P est accessibles gratuitement. Beaucoup d'entreprises essaient de mettre en place un système de P2P qui permettrait de rapporter de l'argent, mais le modèle économique est difficile à trouver, et cela pose de nombreux problèmes techniques.

- Les P2P anonymes : Dans les réseaux P2P classiques, il est possible de savoir que telle adresse IP possède tel fichier. Cela permet donc d'identifier théoriquement qui distribue ou télécharge un fichier. Les P2P anonymes sont différents: ils fonctionnent de telle manière que vous ne savez pas qui distribue un fichier (utilisation d'intermédiaires), et il est impossible de savoir ce que vous téléchargez (tout est chiffré).

  Certains P2P non-anonymes peuvent fonctionner avec des surcouches qui permettent de les rendre un peu plus anonymes, mais au prix de pertes de performances importantes.

- Les P2P privés : Ce système consiste à créer des réseaux P2P privés, uniquement entre amis. Vous ne pouvez accéder à un réseau P2P privé que sur invitation. Chacun peut créer un réseau privé, et se relier à d'autres réseaux privés. Toute personne extérieur au réseau P2P privé ne peut pas voir ce qu'il contient, ni ce que les personnes échangent. Ils sont encore assez peu développés, mais ils fonctionnent bien et ne possèdent pas les inconvénients des P2P anonymes. [8]

I.4 Modèle client serveur

I.4.1 Définition

L'architecture client-serveur est un modèle de fonctionnement logiciel qui se réalise sur tout type d'architecture matérielle (petites à grosses machines), à partir du moment où ces architectures peuvent être interconnectées. [9]

Le dialogue entre les applications client-serveur peut se résumer par :

Le serveur : est un programme offrant un service sur un réseau (machine offrant un service). Il assure la gestion des données partagées entre les utilisateurs.

Le client : est un programme qui émet des requêtes (demande de service). Il gère l'interface graphique de la station de travail personnelle d'un utilisateur.

Application client-serveur : c'est une application qui fait appel à des services distants à travers d'un échange de messages (les requêtes) plutôt que par un partage de données (mémoire ou fichiers). Le client est toujours l'initiateur du dialogue. [4]

Le modèle client-serveur possède quatre types fondamentales qui sont :

- Client-serveur traditionnel (Remote Procedure Call).
- Client-serveur objet (Remote Method Invocation).
- Client-serveur de données (via requêtes SQL).
- Client-serveur Web (via requêtes HTTP).

I.4.2 Client-serveur traditionnel

I.4.2.1 Définition Remote procedure call

En informatique et en télécommunication, **RPC** (Remote Procedure Call) est un protocole réseau permettant de faire des appels de procédures sur un ordinateur distant à l'aide d'un serveur d'applications. Ce protocole est utilisé dans le modèle client-serveur pour assurer la communication entre le client, le serveur et des éventuels intermédiaires. [10]

I.4.2.2 Propriétés des RPC

Modèle au niveau du langage évolué de l'appel de fonction

- Utilisation de la programmation structurée
- Facile à comprendre pour le programmeur

Interaction Demande/ réponse synchrone

- Mécanisme habituée par les programmeurs
- Des réponses correspondent aux demandes

- construit dans la synchronisation des demandes et des réponses

La transparence de la distribution (dans le cas de non-échec)

- masquer la complexité d'un système distribué,

Diverses garanties de fiabilité

Faire face à certaines pannes des systèmes distribués. [11]

I.4.3 Client-serveur objet

I.4.3.1 Définition Remote Method Invocation

RMI (*Remote Method Invocation*) est une API Java permettant de manipuler des objets distants (c'est-à-dire un objet instancié sur une autre machine virtuelle, éventuellement sur une autre machine du réseau) de manière transparente pour l'utilisateur, c'est-à-dire de la même façon que si l'objet était sur la machine virtuelle (JVM) de la machine locale. [12]

I.4.3.2 Architecture de RMI

L'architecture de RMI est schématisée ci-dessous :

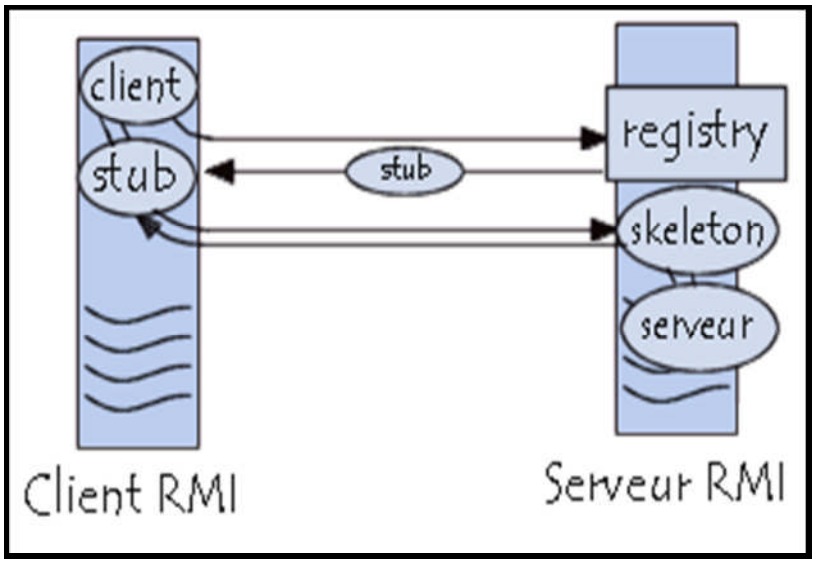

Figure I.1 :L'architecture de RMI [12].

Lorsqu'un objet instancié sur une machine cliente désire accèder à des méthodes d'un objet distant, il effectue les opérations suivantes :

- il localise l'objet distant grâce à un service de désignation : le **registre RMI**

- il obtient dynamiquement une image virtuelle de l'objet distant (appelée **stub** ou *souche* en français).

  Le stub possède exactement la même interface que l'objet distant. [13]

I.4.4 Client-serveur de donnée

I.4.4.1 Définition

Dans une architecture client-serveur, un applicatif est constitué de trois parties : l'interface utilisateur, la logique des traitements et la gestion des données. Le client n'exécute que l'interface utilisateur et la logique des traitements, laissant au serveur de bases de données la gestion complète des manipulations de données.

- Le serveur de bases de données fournit des services aux processus clients. Les tâches qu'il doit prendre en compte sont : la gestion d'une mémoire cache, l'exécution de requêtes exprimées en SQL, exécuter des requêtes mémorisées, la gestion des transactions, la sécurité des données.
- Le client doit ouvrir une connexion pour pouvoir profiter des services du serveur. Il peut ouvrir plusieurs connections simultanées sur plusieurs serveurs. Le client peut soumettre plusieurs requêtes simultanément au serveur et traiter les résultats de façon asynchrone. [14]

I.4.4.2 Le modèle relationnel

Le modèle relationnel est basé sur une organisation des données sous forme de tables. La manipulation des données se fait selon le concept mathématique de relation de la théorie des ensembles, c'est-à-dire l'algèbre relationnelle. L'algèbre relationnelle a été inventée en 1970 par E.F. Codd, le directeur de recherche du centre IBM de San José. Elle est constituée d'un ensemble d'opérations formelles sur les relations. Les opérations relationnelles permettent de créer une nouvelle relation (table) à partir d'opérations élémentaires sur d'autres tables (par exemple l'union, l'intersection, ou encore la différence). [15]

I.4.5 Client-serveur web

I.4.5.1 description

La consultation de documents sur le Web s'appuie au départ sur une architecture informatique centralisée dans laquelle les rôles sont clairement définis. L'architecture de Web est de type client-serveur :

- Un logiciel serveur (le démon HTTP ou serveur Web) met à disposition des informations.
- Un logiciel client (le navigateur Web) se connecte à ce serveur pour demander à recevoir ces informations qui seront ensuite affichées sur l'écran de l'utilisateur.

Le processus de communication s'effectue donc toujours de manière asymétrique selon la règle suivante : c'est le navigateur qui demande des informations au serveur qui les lui transmet en retour. Le Web est avant tout un protocole de diffusion d'information qui pourrait sembler inadapté au développement d'application. L'apparente simplicité du protocole sur lequel il est bâti est pourtant la principale raison de son succès. [4]

I.5 Outils de travail

Dans notre projet les outils utilisés sont : Java, NetBeans, EasyPHP, MySQL et Hibernate.

I.5.1 java

I.5.1.1 Définition

C'est un langage de programmation orienté objet, développé par Sun Microsystems. Il permet de créer des logiciels compatibles avec de nombreux systèmes d'exploitation (Windows, Linux, Macintosh, Solaris). Java donne aussi la possibilité de développer des programmes pour téléphones portables et assistants personnels. Enfin, ce langage peut-être utilisé sur internet pour des petites applications intégrées à la page web (applet) ou encore comme langage serveur. [16]

I.5.1.2 Les caractéristiques

Java regroupe toutes les caractéristiques d'un vrai langage professionnel : il est simple tout en étant flexible et puissant, ce n'est pas la syntaxe et la grammaire du langage qui lui confèrent sa portabilité, mais plutôt la machine virtuelle et les librairies :

- Le langage Java est simple, familier et orienté objet
- Le langage Java est distribué
- Le langage Java est interprété
- Le langage Java est robuste et sûr
- Le langage Java est portable et indépendant des plate- formes
- Le langage Java est dynamique et multithread [4]

I.5.1.3 Les IDE java

Les principaux IDE existants aujourd'hui:

- Borland JBuilder X version professionnelle (commercial)
- IBM WebSphere Studio Application Developer 5.1.1 (commercial)
- Sun One Studio 5.1 (commercial)
- Jetbrains Intellij Idea (commercial)
- IBM Eclipse (gratuit)
- Sun Netbeans (gratuit) [17]

I.5.1.4 Les versions de Java

Le langage Java n'est pas fixé dans le temps, il évolue, des ajouts au langage sont régulièrement proposés, et l'API standard s'enrichie continuellement passant de 200 classes à près de 4000 entre la première et la "dernière" version :

- JDK 1.0 (1996) : C'est la première version stable du langage, de ce fait la totalité des navigateurs la supporte
- JDK 1.1 (1997) : Elle apporte des améliorations syntaxiques ainsi que des progrès au niveau de l'interface utilisateur AWT et de la gestion des exceptions.
- J2SE 1.2 (1998) : La version "2" apporte des améliorations multimédias. Elle permet par exemple l'utilisation d'interface utilisateur graphique avancée avec Swing, ainsi que la possibilité d'interagir avec elle par le moyen de glisser-déposer

- J2SE 1.3 (2000) : L'amélioration multimédia se poursuit avec l'apparition de l'API Java Sound.
- J2SE 1.4 (2002) : Multimédia toujours, avec une bibliothèque de gestion d'images.
- J2SE 5.0 (2004) : Ajouts de nouvelles fonctionnalités au langage, comme les annotations, les énumérations, la boucle "for each", les nombre d'arguments variables (vargs), ou la conversion automatique entre les types primitifs et leurs classes enveloppes.
- Java SE 6 (2006)
- Java SE 7 (2011) [18]

I.5.2NetBeans

I.5.2.1 Définition

NetBeans est un environnement de développement intégré (IDE) pour Java, placé en open source par Sun en juin 2000 sous licence CDDL (Common Development and Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++, XML et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages web).

NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X et Open VMS. [19]

I.5.3 EasyPHP

I.5.3.1 Définition

EasyPHP propose un ensemble d'applications permettant de faire fonctionner des scripts PHP en local, c'est-à-dire sans avoir à se connecter à un serveur externe sur

Internet. Il peut tout à fait être comparé aux logiciels WAMP5 ou XAMPP, qui sont des équivalents.

EasyPHP est un environnement de développement composé de deux serveurs : un serveur **Web Apache** et un serveur de **base de données MySQL**. De plus, il possède un interpréteur de scripts (PHP) et d'un module d'administration SQL nommé PhpMyAdmin. [20]

I.5.4 MySQL

I.5.4.1 Définition

MySQL est un serveur de bases de données relationnelles Open Source.

Un serveur de bases de données stocke les données dans des tables séparées plutôt que de tout rassembler dans une seule table. Cela améliore la rapidité et la souplesse de l'ensemble. Les tables sont reliées par des relations définies, qui rendent possible la combinaison de données entre plusieurs tables durant une requête. Le SQL dans "MySQL" signifie "Structured Query Language" : le langage standard pour les traitements de bases de données. [21]

I.5.5 Swing

I.5.5.1Définition

Swing fait partie de la bibliothèque Java Foundation Classes (JFC). C'est une API dont le but est similaire à celui de l'API AWT mais dont les modes de fonctionnement et d'utilisation sont complètement différents. Swing a été intégré au JDK depuis sa version 1.2. [22]

I.5.6 Hibernate

I.5.6.1 Définition

Hibernate est un Framework Java de persistance qui permet de faire correspondre des tables de bases de données relationnelles avec des objets java simples (Exemple : une classe Personne toute simple avec un 'nom', un 'prénom' et une 'adresse' comme attributs).

Il permet au développeur Java de s'affranchir de la connaissance du langage SQL et

de réfléchir en termes d'objet tout en laissant à Hibernate le soin de réaliser toutes les opérations dans la base de données. [23]

I.5.6.2 Architecture de Hibernate

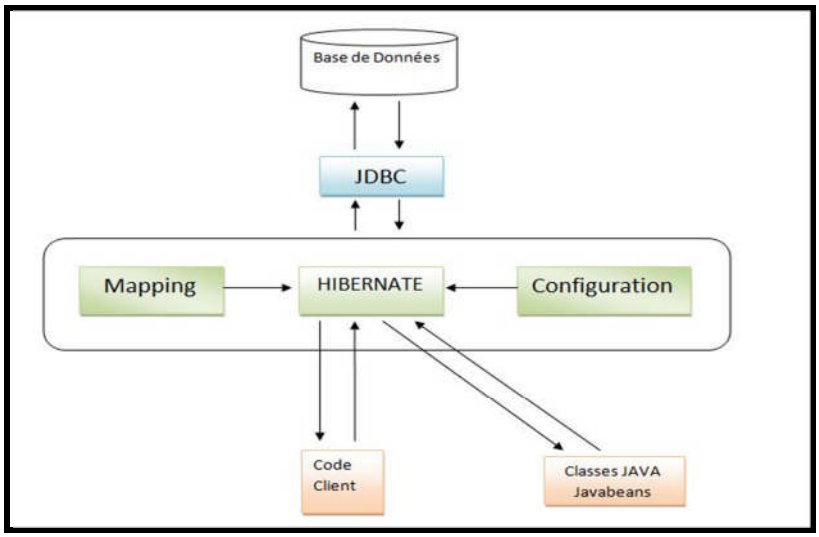

Figure I. 2:une vue (très) haut niveau de l'architecture Hibernate. [23]

I.5.6.3 Les avantage

- Hibernate génère le code SQL nécessaire, ce qui rend l'application plus portable (s'adapte à la base de données)
- La persistance est transparente. Vous pouvez faire de vos classes métiers des classes persistantes sans ajout de code.
- La récupération de données est optimisée. On peut interroger la base de données de plusieurs façons (Requête SQL, langage HQL...)
- Portabilité du code en cas de changement de la base de données

I.5.6.4 Les inconvénients

- Il est dure de faire des requêtes complexe avec HQL (ex: sous requêtes)
- Etant une technologie jeune, il reste des problèmes à résoudre (ex: les fichiers de mapping ne sont pas toujours bien formés)

- Ne se base pas sur les standards. [24]

I.6 Conclusion

Nous avons présenté dans ce chapitre la programmation réseau où nous avons basé sur le modèle diffusion, Peer to Peer et client-serveur. Les outils utilisés dans notre PFE sont : Java qui nécessite un environnement de développement intégré comme NetBeans, EasyPHP, MySQL et Hibernate.

Dans le chapitre suivant nous présenterons en détail les différentes étapes nécessaires pour la réalisation de notre application.

CHAPITRE II :

PRÉSENTATION DE L'APPLICATION

II.1 Introduction :

Dans ce chapitre nous allons présenter les étapes nécessaires pour la réalisation de notre projet « Gestion d'une bibliothèque » qui permet de chercher un livre avec son code, titre ou auteur.

II.2 Présentation de l'application :

II.2.1 Création de la base de données :

Nous avons créé une base de données « fac » et une table « bibliothèque » à l'intérieur la figure suivante montre le résultat obtenue.

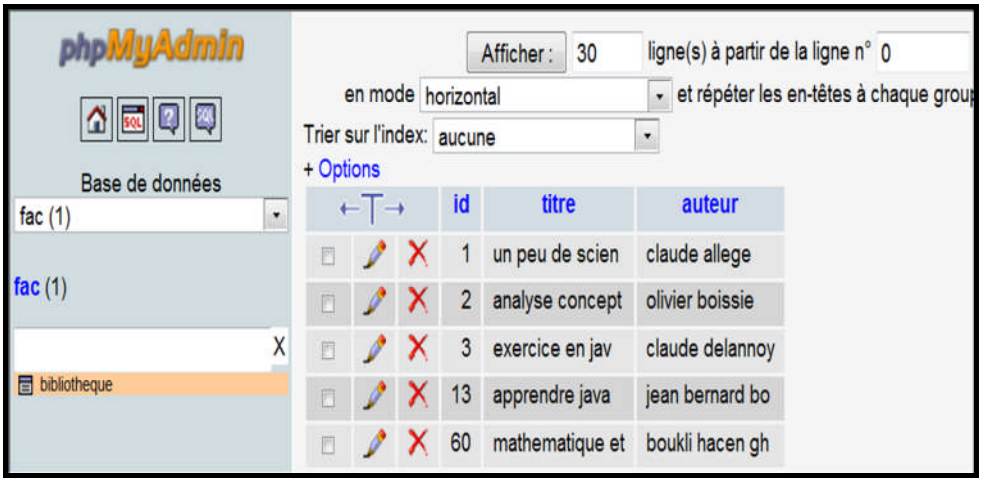

Figure II.1: La table bibliothèque.

II.2.2 Accéder à notre base de données à partir de NetBeans :

- Lancer NetBeans IDE, cliquer sur **« Services »** et choisir **« Database »** après **« Drivers »**.Voir figure 4

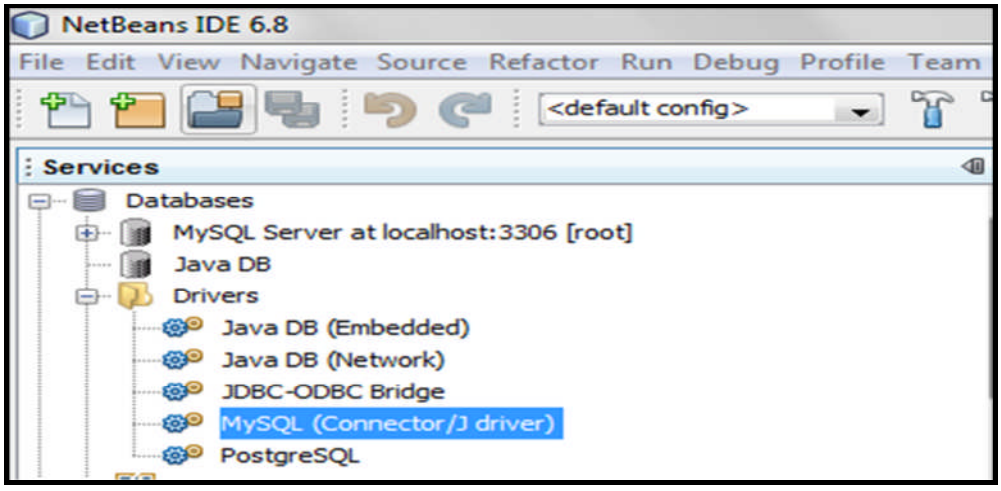

Figure II.2: Accès à notre base de données.

Clique-droit sur « **MySQL (Connector/J driver**) » et choisir **« Connect Using…»**

Saisir le nom de la base de données dans le champ « **Database** », cliquez sur « **OK** ».

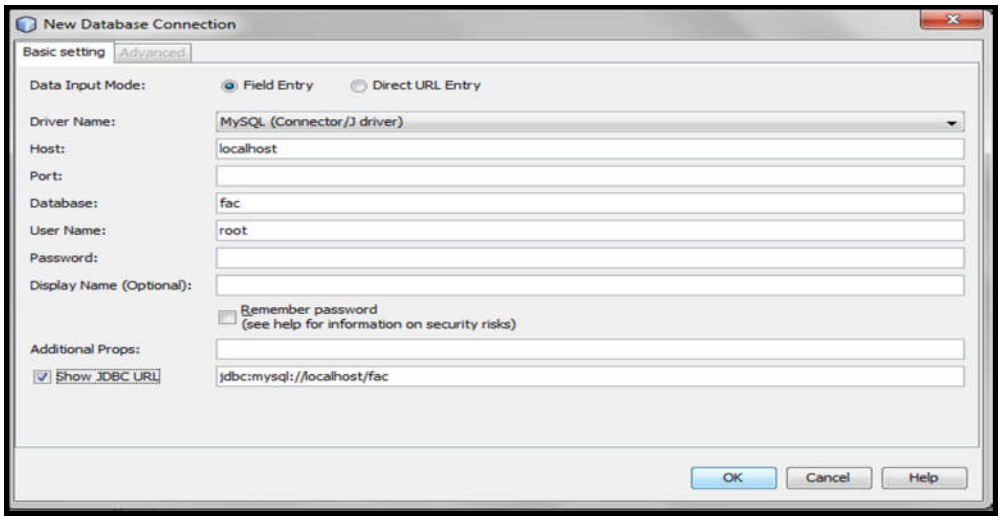

Figure II.3: Se connecter à la base de données.

- Notre base de données est maintenant connectée, pour afficher la table clique-droit sur « **bibliotheque** » en suite choisir « **View Data…** »

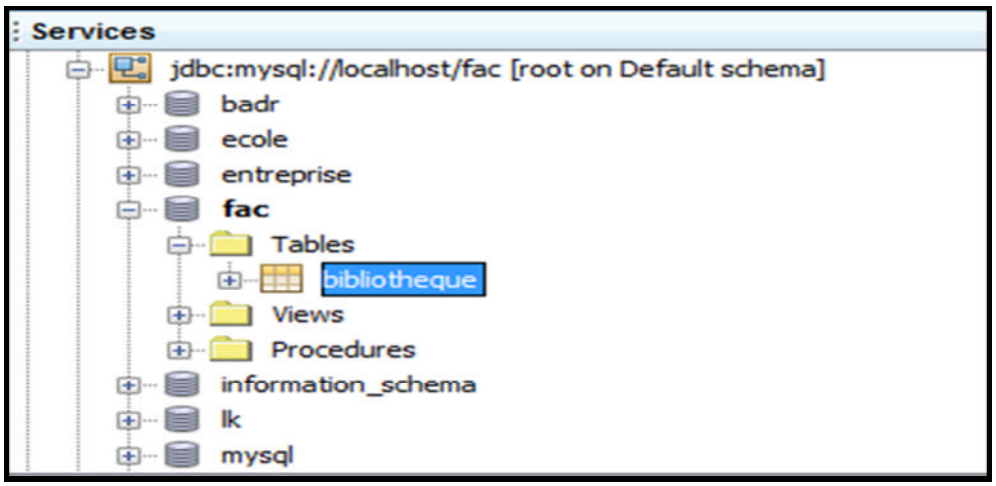

Figure II.4: Affichage de la table sous NetBeans.

- Les données s'affiche comme suit :

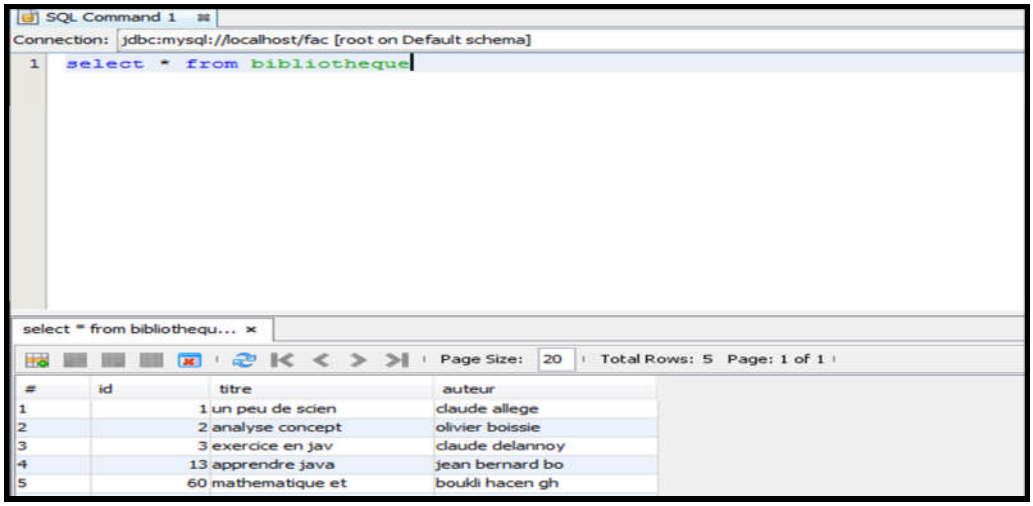

Figure II.5: Résultat de l'affichage de la table.

II.2.3 Création du projet :

II.2.3.1 Créez un projet Swing :

- Cliquez sur **« File »** choisir **« New Project »,** prendre **« Java Application »** de catégories **« Java »** et cliquez sur **« Next ».**
- Saisir le nom du projet, pour notre projet c'est « **BibliothequeHib** », désélectionner « **Create Main Class** », cliquez sur « **Finish** ». Voir la figure suivante

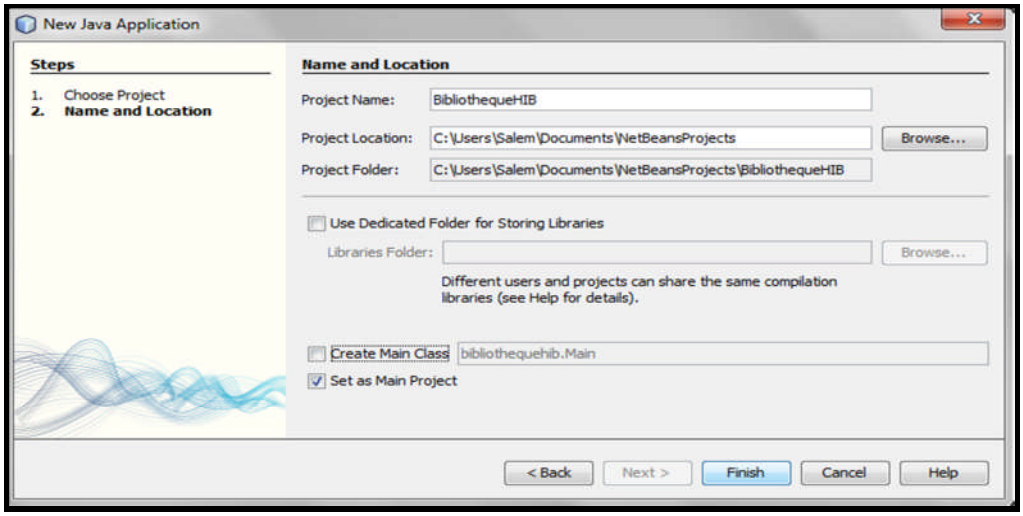

Figure II.6: Créer le projet.

II.2.3.2 L'ajout du Framework « Hibernate » dans la Librairie :

- Clique-droit sur « libraries » et appuyez sur « **Add Library** » et choisir « **Hibernate** ».

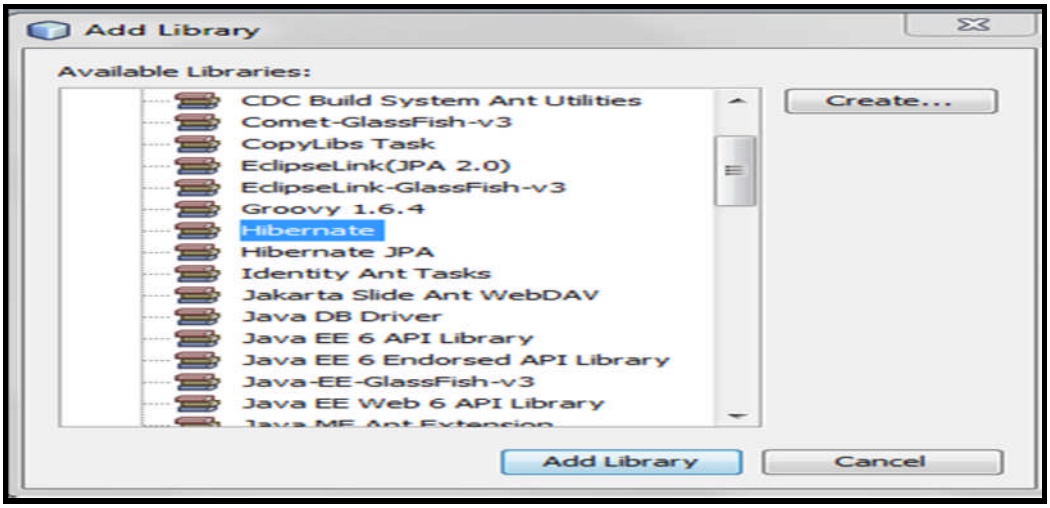

Figure II.7: Ajout la librairie Hibernate.

II.2.3.3 Créer le fichier de configuration hibernate.cfg.xml :

- Clique-droit sur « **Source Packages** », choisir « **New** » appuyez sur « **Other** ».
- Sélectionnez « **Hibernate Configuration Wizard** » de catégories **« Hibernate** ».

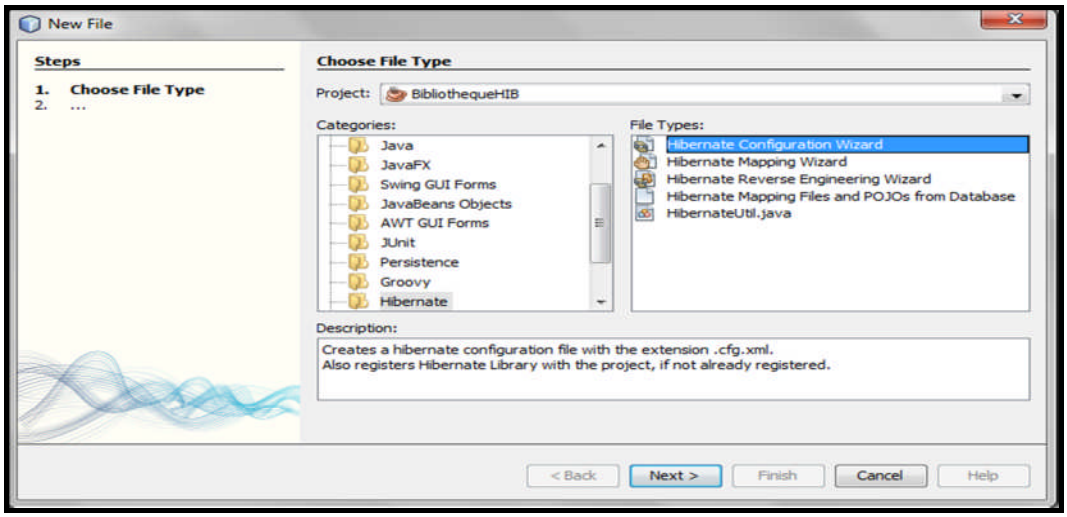

Figure II.8: Créer le fichier de configuration (1/3).

- Clique sur « **Next** ».
- S'affiche :

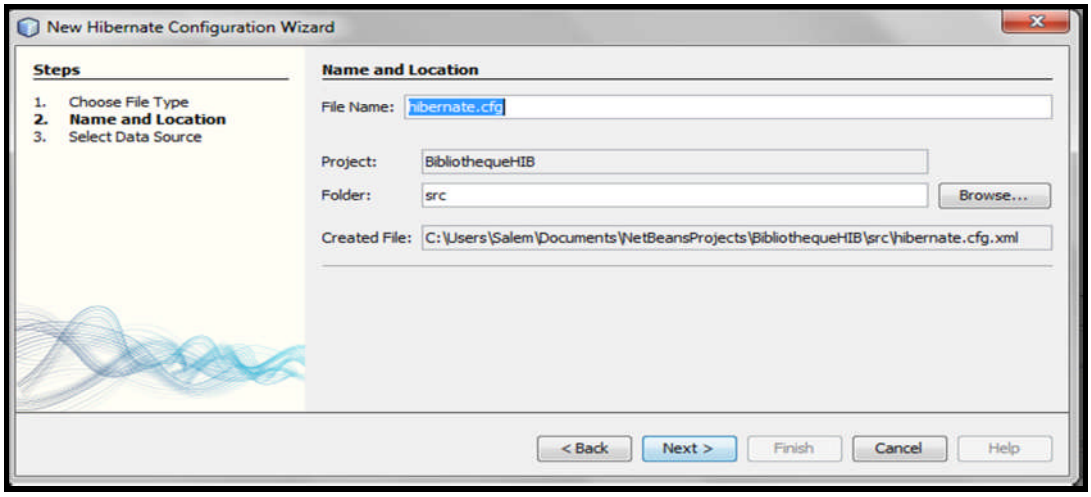

Figure II.9: Créer le fichier de configuration (2/3).

- Cliquez sur « Next ».

- Sélectionnez la base de données « **fac** ».
- Cliquez sur « **Finish** ».

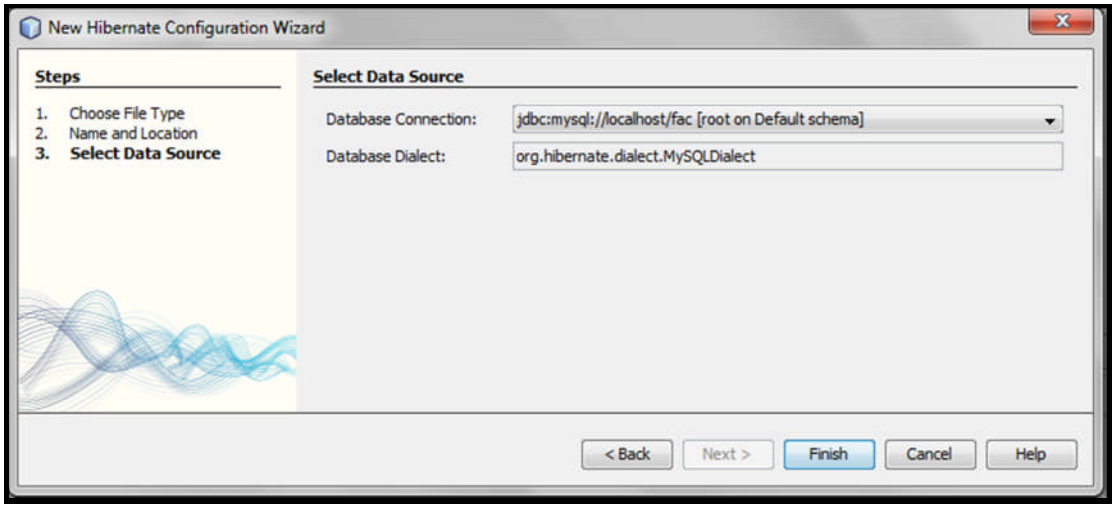

Figure II.10: Créer le fichier de configuration (3/3).

- S'affiche : hibernate.cfg.xml : Contenu **Design**

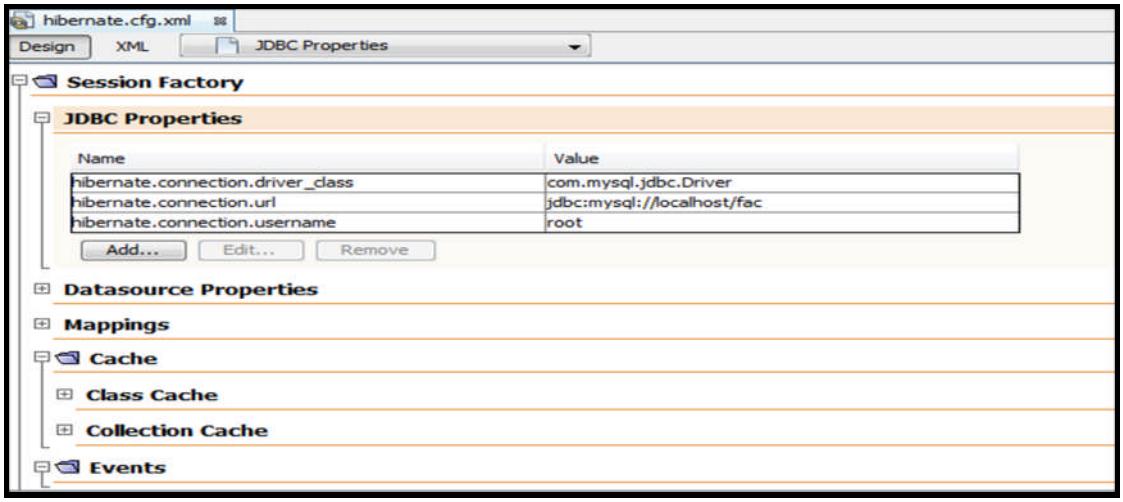

Figure II.11: Contenu de design de fichier de configuration.

- hibernate.cfg.xml: Contenu **XML**

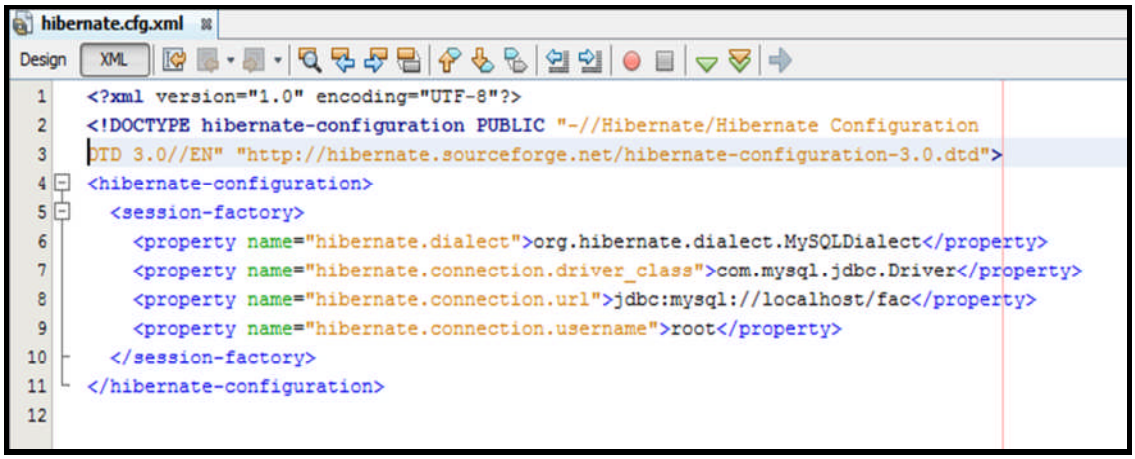

Figure II.12: Contenu XML de fichier de configuration.

II.2.3.4Modifier hibernate.cfg.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate
Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-
configuration-3.0.dtd">
<hibernate-configuration>
  <session-factory>
    <property
name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
    <property
name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
```

```
    <property
name="hibernate.connection.url">jdbc:mysql://localhost/fac</property>
    <property name="hibernate.connection.username">root</property>
<property name="hibernate.show_sql">true</property>
<property
name="hibernate.query.factory_class">org.hibernate.hql.classic.ClassicQuery
Translator Factory</property>
  </session-factory>
</hibernate-configuration>
```

- **hibernate.show_sql** (true): permet d'afficher les requêtes dans la console de l'IDE.
- **hibernate.query.factory_class**: choisi l'implémentation du parseur de requête.

II.2.3.5 Créer le « Helper File» : fichier HibernateUtil.java :

- Clique-droit sur « **Source Packages** », sélectionnez « **New** » appuyez sur « **Other** ».
- Sélectionnez « **Hibernate** » de catégories et « **HibernateUtil.java** » de File Types.
- Cliquez « **Next** ».

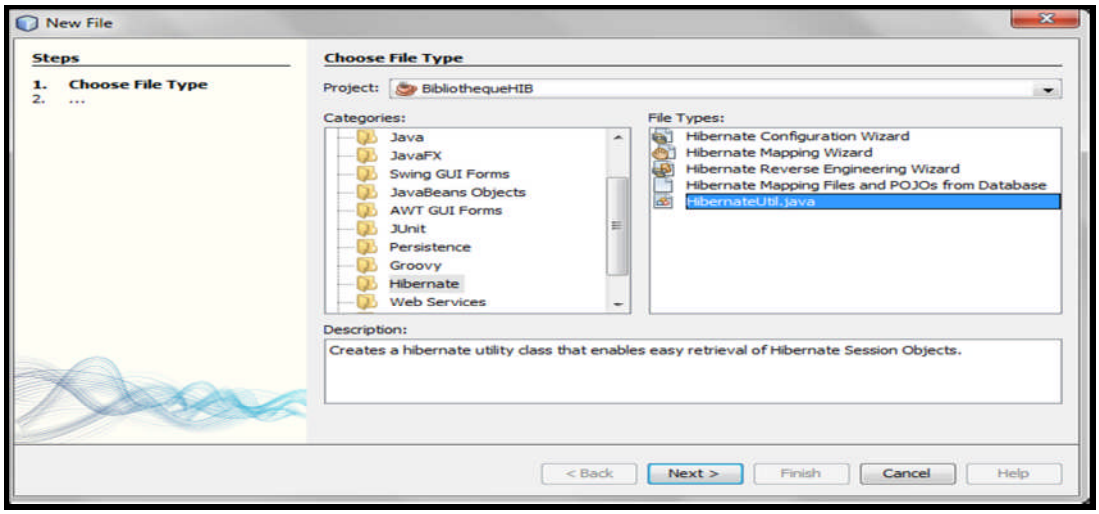

Figure II.13: Créer le Helper File (1/2).

- Saisir « **HibernateUtil** » pour le nom de la classe, saisir « **bibliotheque.util** » comme nom du package, cliquez sur « **Finish** ».

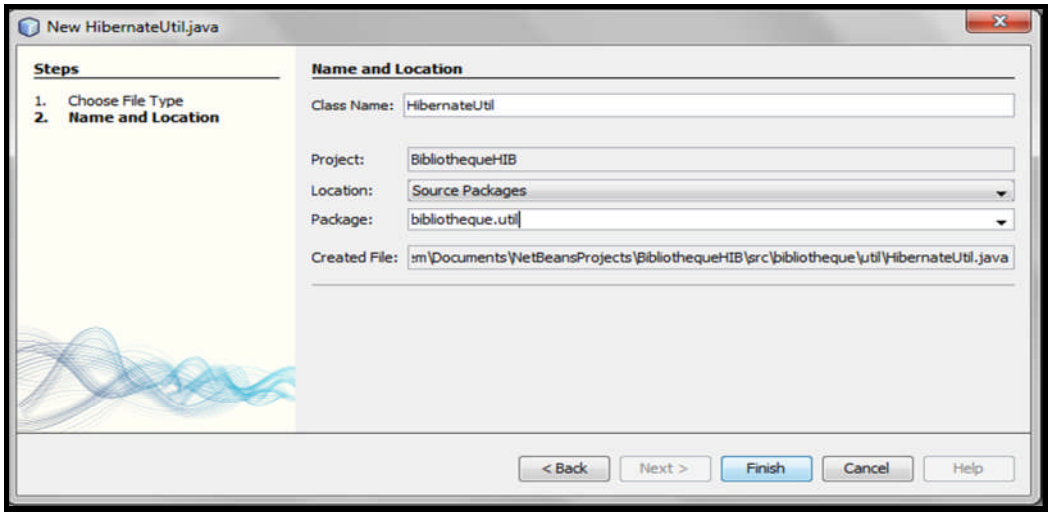

Figure II.14: Créer le Helper File (2/2).

- **Ci-dessous la description du « helper » :**

```
package bibliotheque.util;
import org.hibernate.cfg.AnnotationConfiguration;
import org.hibernate.SessionFactory;
public class HibernateUtil {
    private static final SessionFactory sessionFactory;
    static {
        try {
            sessionFactory = new
AnnotationConfiguration().configure().buildSessionFactory();
        } catch (Throwable ex) {
            System.err.println("Initial SessionFactory creation failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }
    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}
```

II.2.3.6 Générer les fichiers de mapping et les classes java :

- Pour mapper « **Bibliotheque.java** » à la table « **bibliotheque** », on doit passer par deux étapes :
- **Hibernate Reverse Engineering Wizard (.xml).**
- **Hibernate Mapping Files (.xml) and POJOs from a Database (.java).**

II.2.3.7 Créer le fichier « Reverse Engineering » :

- Clique-droit sur « **Source Packages** », sélectionnez « **New** » appuyez sur « **Other** ».

- Sélectionnez « **Hibernate** » de catégories et « **Hibernate Reverse Engineering Wizard** » de File Types.
- Cliquez sur « **Next** ».

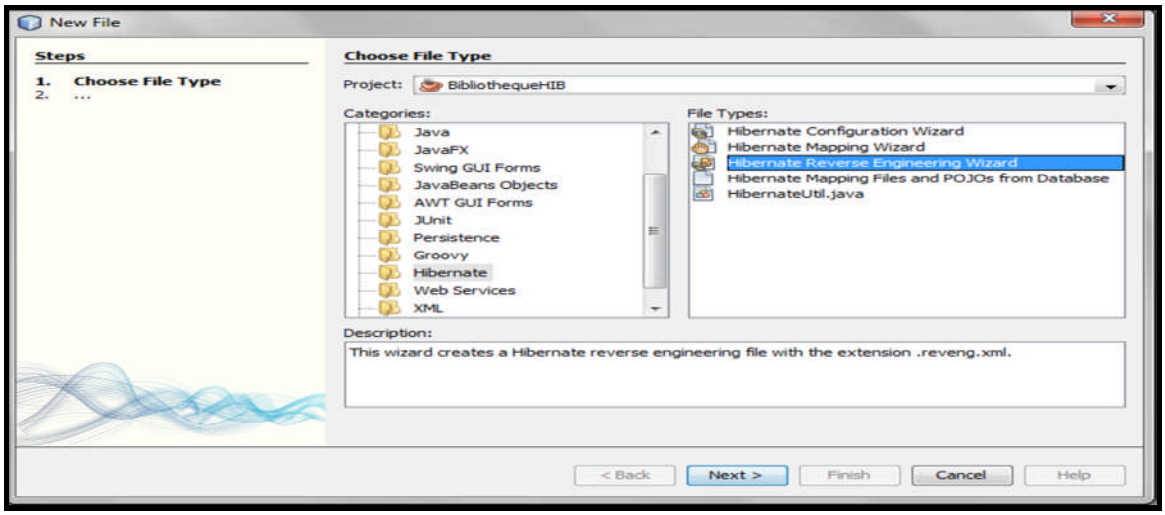

Figure II.15: Créer le fichier Reverse Engineering (1/2).

- Ensuite :
- Saisir « **hibernate.reveng** » comme nom.
- Laissez « **src** » comme Folder.
- Cliquez « **Next** ».

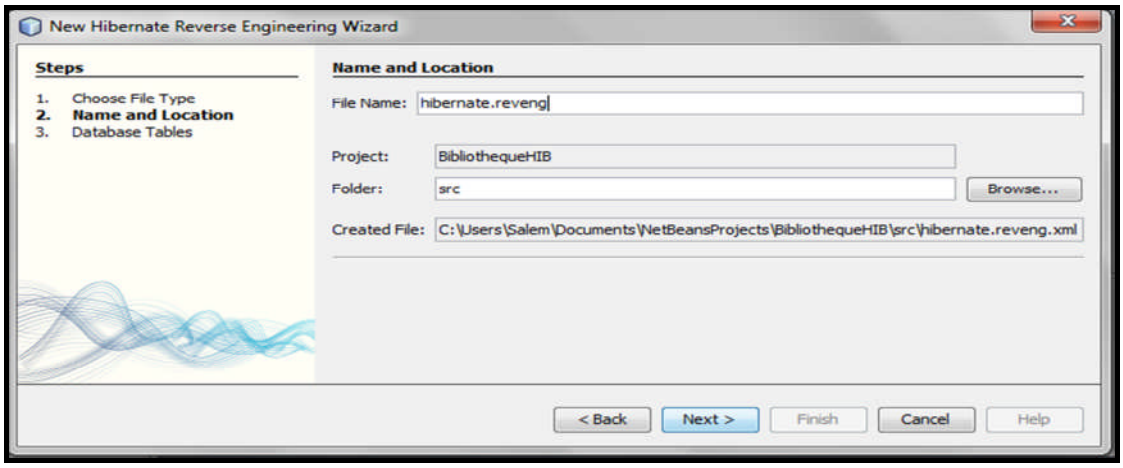

Figure II.16: Créer le fichier Reverse Engineering (2/2).

- S'affiche :

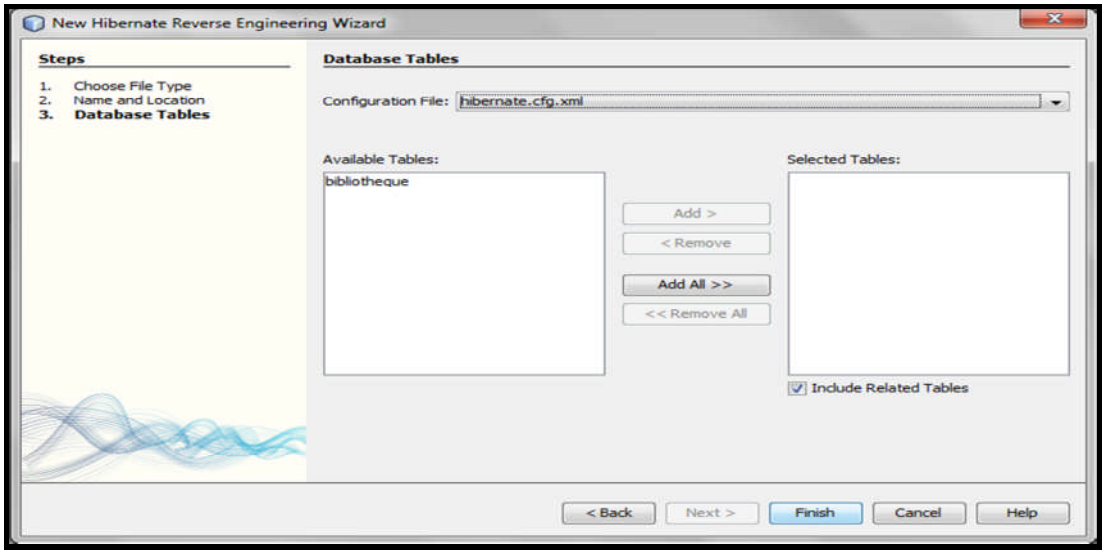

Figure II.17: Sélection de la table bibliothèque.

- Sélectionnez la table « **bibliotheque** », choisir « **Add** » et cliquez sur « **Finish** ».

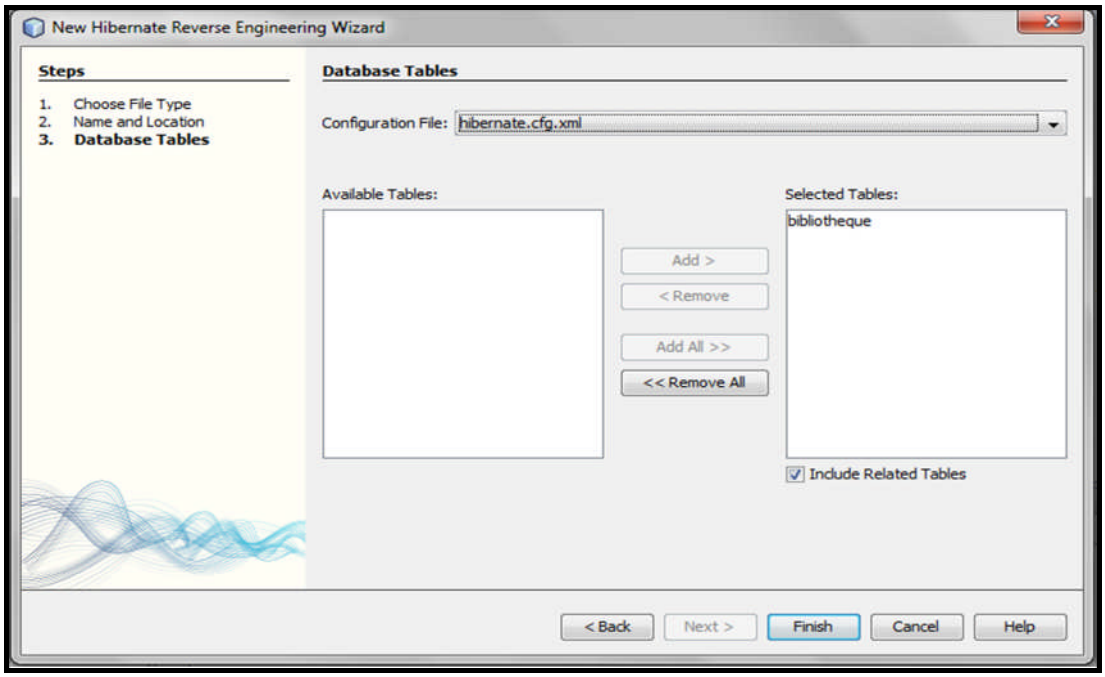

Figure II.18: Ajouter la table bibliothèque.

- Ci-dessous le contenu du **hibernate.reveng.xml** :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-reverse-engineering PUBLIC "-//Hibernate/Hibernate
 Reverse Engineering DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-
reverse-engineering-3.0.dtd">
<hibernate-reverse-engineering>
  <schema-selection match-catalog="fac"/>
  <table-filter match-name="bibliotheque"/>
```

```
</hibernate-reverse-engineering>
```

II.2.3.8 Créer « Hibernate Mapping Files and POJOs from a Database »:

- Clique-droit sur « **Source Packages** », sélectionnez « **New** » appuyez sur « **Other** ».
- Sélectionnez « **Hibernate** » de catégories et « **Mapping Files and POJOs from a Database**» de Files Types.
- Cliquez sur « **Next** ».

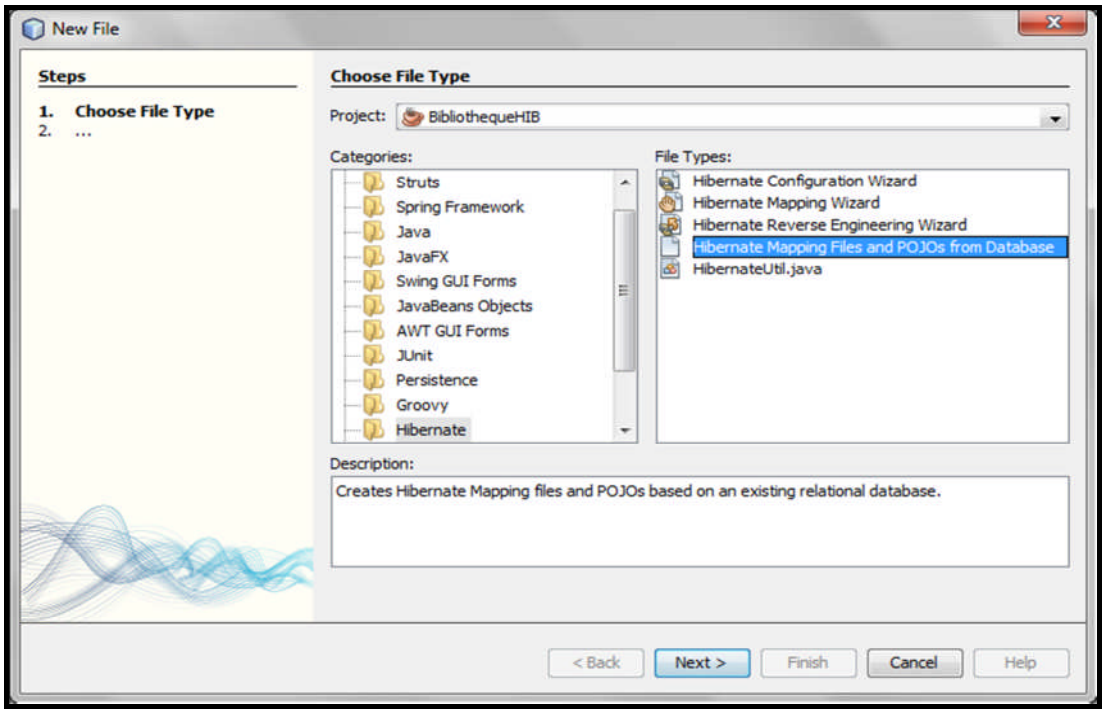

Figure II.19: Hibernate Mapping Files and POJOs From a Database (1/2).

- Sélectionnez «**hibernate.cfg.xml**» de «**Hibernate Configuration File**».
- Sélectionnez «**hibernate.reveng.xml** » de «**Hibernate Reverse Engineering File**».
- Assurer que «**Domain Code**» et « **Hibernate XML Mappings** » sont cochés.
- Saisir «**bibliotheque.entity** » comme nom de package, cliquez « **Finish** ».

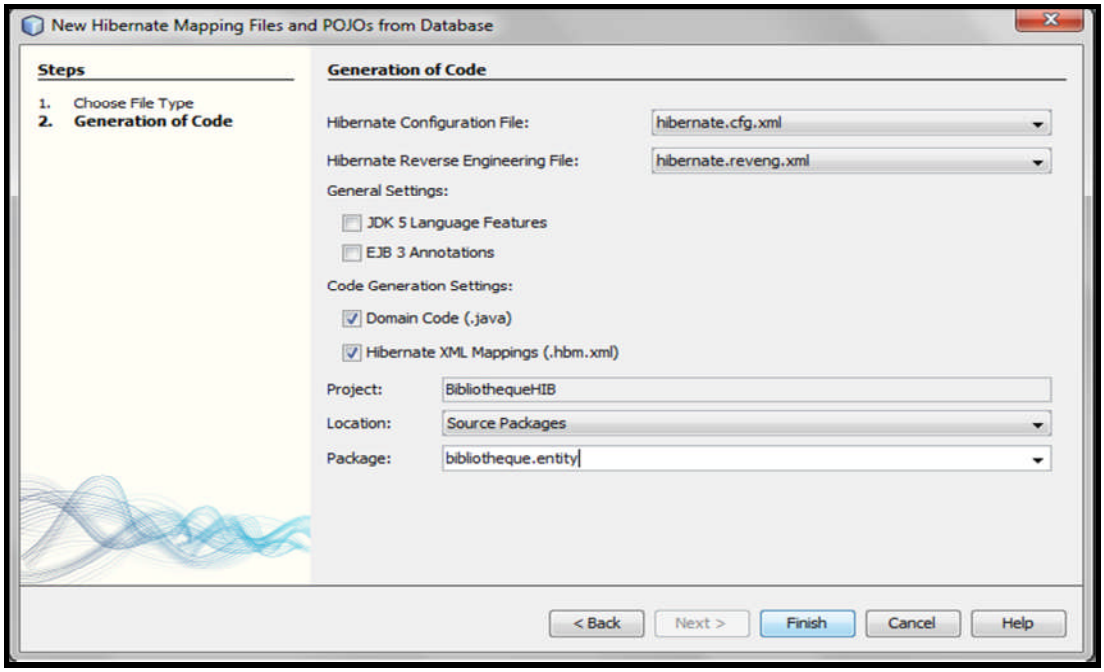

Figure II.20: Hibernate Mapping Files and POJOs From a Database (2/2).

- Ci-dessous l'arborescence de notre projet:

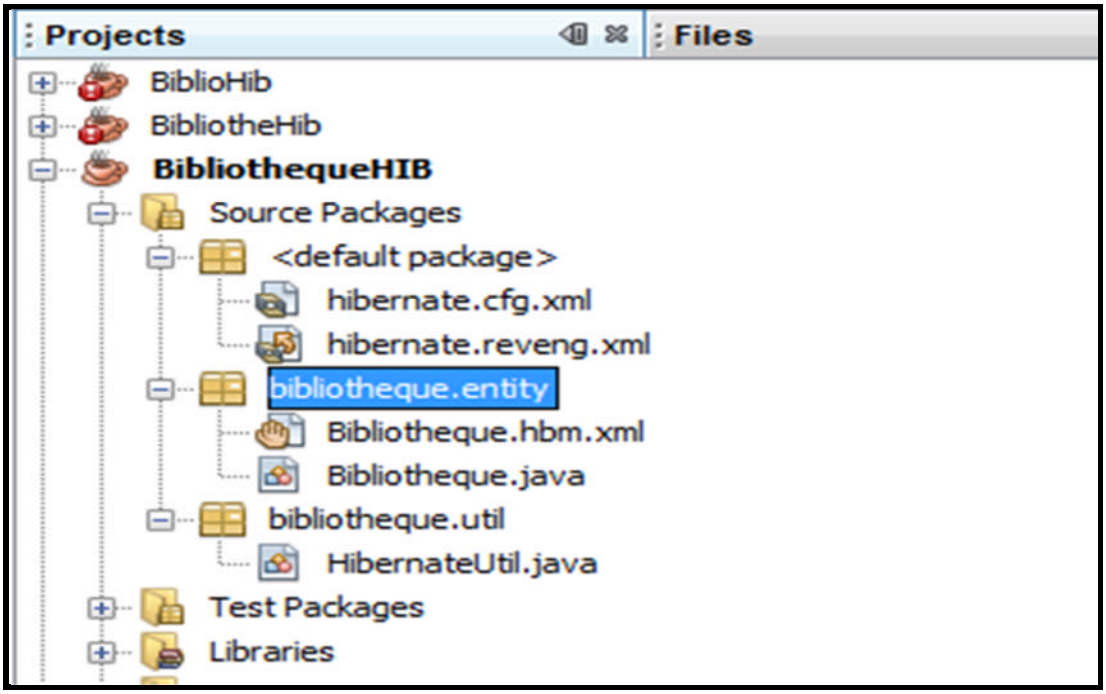

Figure II.21: Arborescence du projet.

- **Bibliotheque.java (POJO) :**

```
package bibliotheque.entity;
public class Bibliotheque  implements java.io.Serializable {
 private int id;
 private String titre;
```

```
 private String auteur;
 public Bibliotheque() {
    }
public Bibliotheque(int id, String titre, String auteur) {
       this.id = id;
       this.titre = titre;
       this.auteur = auteur;
    }
   public int getId() {
        return this.id;
    }
  public void setId(int id) {
        this.id = id;
    }
    public String getTitre() {
        return this.titre;
    }
   public void setTitre(String titre) {
        this.titre = titre;
    }
    public String getAuteur() {
        return this.auteur;
    }
    public void setAuteur(String auteur) {
this.auteur = auteur;
}
}
```

- **Bibliotheque.hbm.xml (le fichier de mapping) :**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping
DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<!-- Generated 11 mars 2014 00:26:00 by Hibernate Tools 3.2.1.GA -->
<hibernate-mapping>
  <class catalog="fac" name="bibliotheque.entity.Bibliotheque" table="bibliotheque">
    <id name="id" type="int">
      <column name="id"/>
      <generator class="assigned"/>
    </id>
    <property name="titre" type="string">
      <column length="30" name="titre" not-null="true"/>
    </property>
    <property name="auteur" type="string">
      <column length="30" name="auteur" not-null="true"/>
    </property>
  </class>
</hibernate-mapping>
```

II.2.3.9 Créer une JFrame pour faire des recherches dans la table des livres de bibliothèque :

II.2.3.10 Créer la JFrame :

- Clique-droit sur le projet, choisir « **New** » appuyez sur « **Other** », sélectionnez « **JFrame Form** » de la catégorie « **Swing GUI Forms** », cliquez sur **« Next** ».

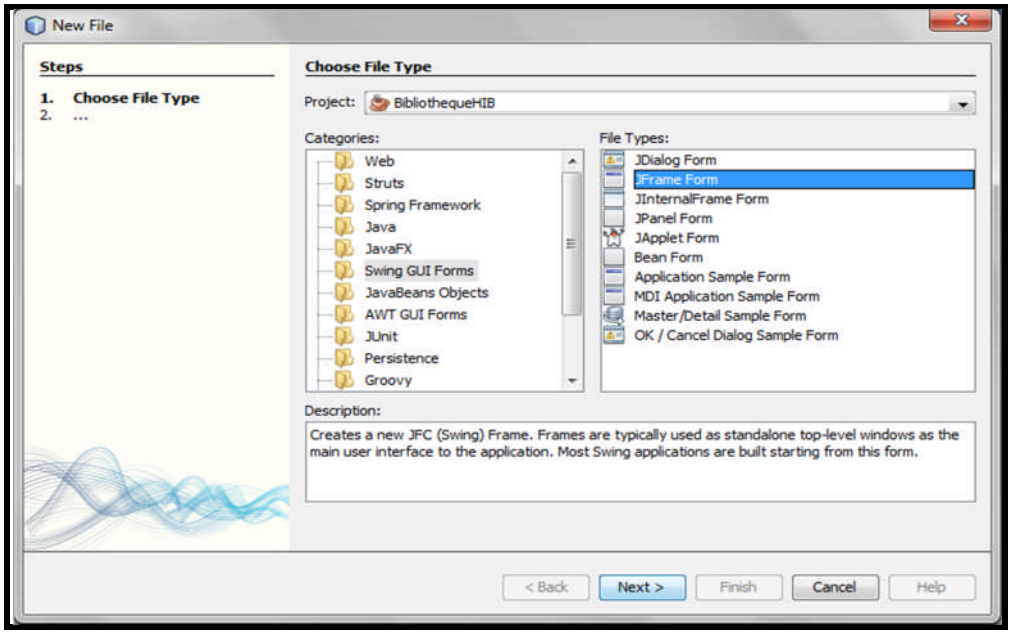

Figure II.22: Créer la JFrame (1/2).

- Ensuite, saisie «**BibliothequeIHM**» comme nom de la classe, et «**bibliotheque.ui**» pour le package, cliquez sur « **Finish** ».

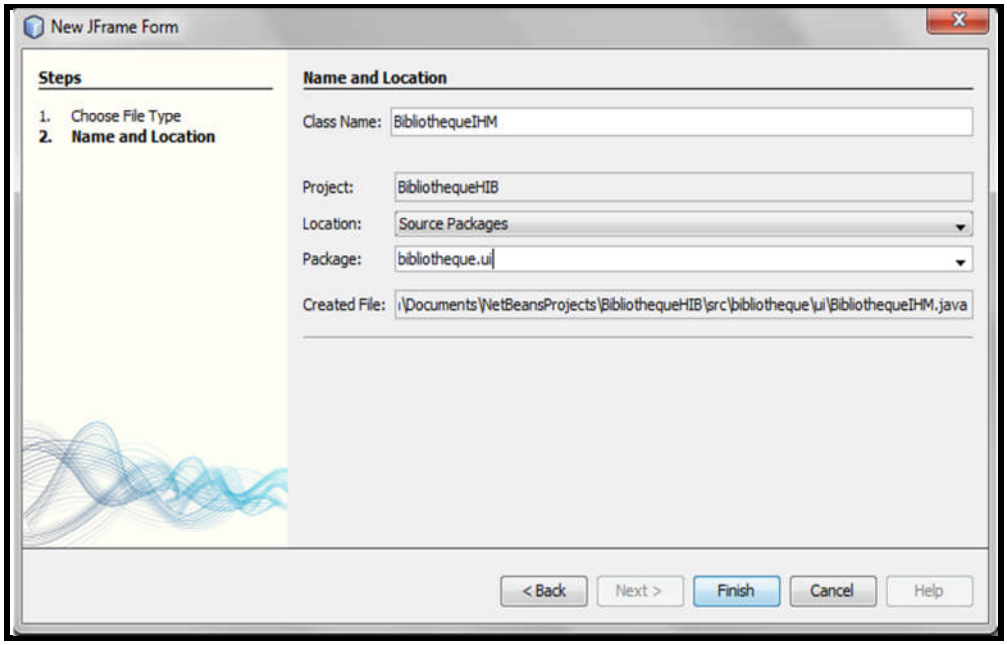

Figure II.23: Créer la JFrame (2/2).

- S'affiche :

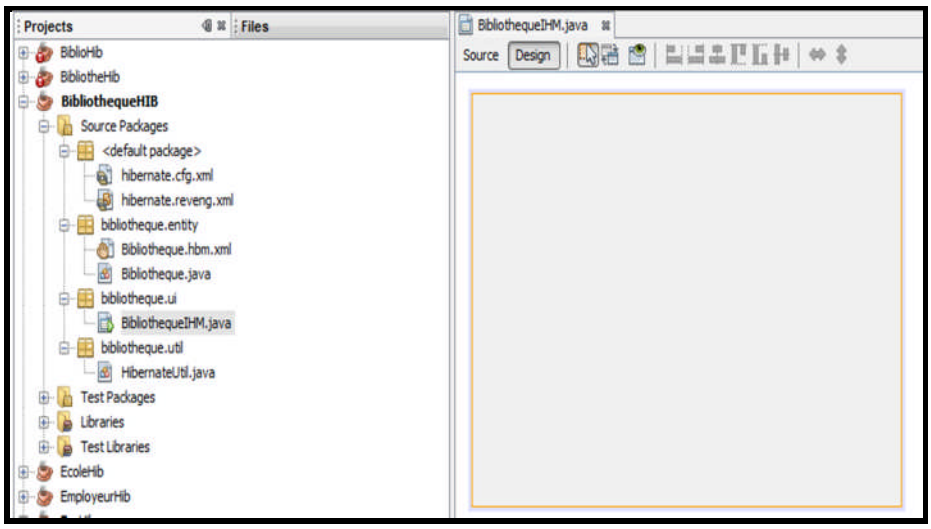

Figure II.24: Zone de la JFrame.

II.2.3.11 Ajouter des éléments à la JFrame :

- Clique-droit sur la forme graphique :
- **Add Form Palette** choisir **Swing Controls** prendre **Label**.
- Changer le texte à: « **merci bien de sélectionner un critère de recherche** ».
- Clique-droit sur la forme graphique:
- **Add From Palette** choisir **Swing Controls** prendre **Radio Button.**
- Saisir: « **recherche par code** »
- Clique-droit sur la forme graphique:
- **Add From Palette** choisir **Swing Controls** prendre **Radio Button.**
- Saisit: « **recherche par titre** »
- Clique-droit sur la forme graphique:
- **Add From Palette** choisir **Swing Controls** prendre **Radio Button.**
- Saisir: « **recherche par auteur** »
- Clique-droit sur la forme graphique:
- **Add Frome Palette** choisir **Swing Controls** prendre **Button Group.**
- Clique-droit sur la forme graphique:
- **Add From Palette** choisir **Swing Controls** prendre **Text Field.**
- **Supprimer le texte par défaut.**
- Clique-droit sur la forme graphique :
- **Add From Palette** choisir **Swing Controls** prendre **Button.**
- Modifier le texte à « **rechercher** ».

- Clique-droit sur la forme graphique :
- **Add From Palette** choisir **Swing Controls** prendre **Table.**

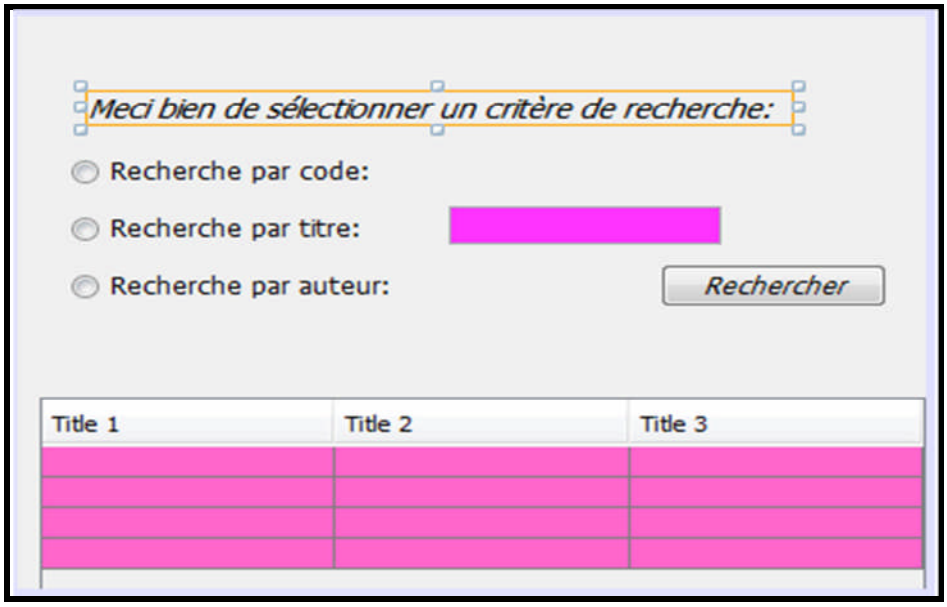

Figure II.25: Ajouter les éléments à la JFrame.

- Clique-droit sur la table choisir **Table Contents** appuyez sur **Column** changez le **Count (3)** et cliquez sur **Close**.

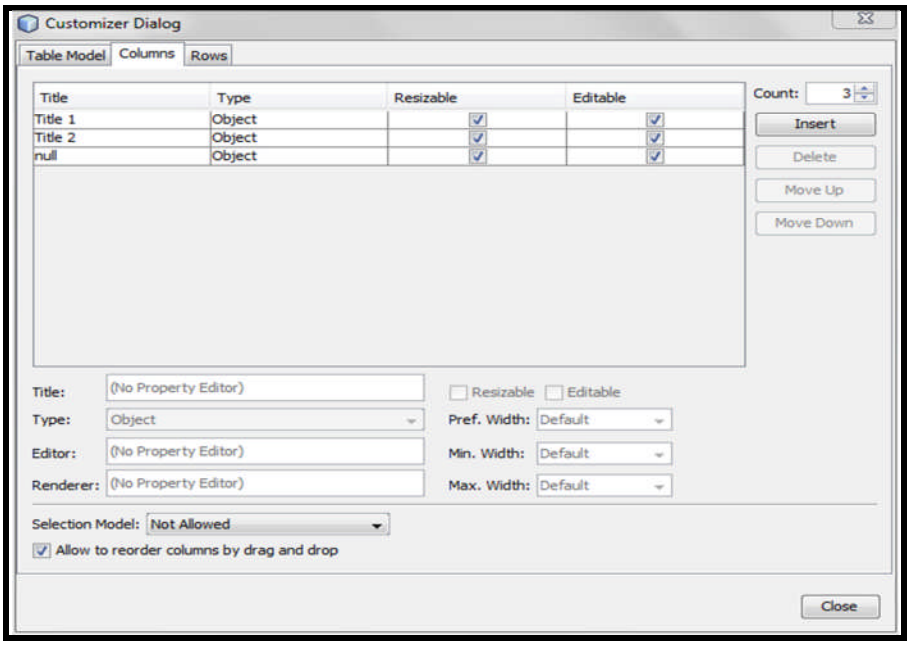

Figure II.26: Modifier la taille de tableau.

II.2.3.12 Changeant le nom des variables :

- La table, le bouton, la zone de texte.
- Pour chaque élément :

- Clique-droit choisir « **Change Variable Name** ».
- La table renomer **resultTable**.
- Le bouton renomer **queryButton**.
- La zone de texte renomer **NameTextField**.

II.2.3.13 Ajouter les déclarations suivantes au code source :

```
private static String QUERY_BASED_ON_NAME="select from Bibliotheque a where a.id like '";
private static String QUERY_BASED_ON_NAME2=" select from Bibliotheque b where b.titre like '";
private static String QUERY_BASED_ON_NAME3=" select from Bibliotheque c where c.auteur like '";
```

II.2.3.14 Ajouter les méthodes suivantes, qui traitent le caractère saisi à partir de l'IHM :

```
private void rechercheparid(){
executeHQLQuery(QUERY_BASED_ON_NAME1+ NameTextField.getText() + "%'");
}
private void recherchepartitre(){
executeHQLQuery(QUERY_BASED_ON_NAME2+ NameTextField.getText() + "%'");
}
private void rechercheparauteur(){
executeHQLQuery(QUERY_BASED_ON_NAME3 + NameTextField.getText() + "%'");
}
```

II.2.3.15 Ajouter la méthode executeHQLQuery():

```
private void executeHQLQuery(String hql) {
    try {
        Session session = HibernateUtil.getSessionFactory().openSession();
        session.beginTransaction(); //Démarrer la transaction
        Query q = session.createQuery(hql);
        List resultList = q.list(); // Exécuter la requête et récupérer le résultat sous forme de List
        displayResult(resultList);
        session.getTransaction().commit(); //Récupérer la transaction et la valider
    } catch (HibernateException he) {
        he.printStackTrace();
    }
}
```

II.2.3.16 Ajouter les imports nécessaires :

- import bibliotheque.util.HibernateUtil;

- import java.util.List;
- import org.hibernate.HibernateException;
- import org.hibernate.Query;
- import org.hibernate.Session;

II.2.3.17 Basculer vers Design et double clic sur le bouton Rechercher :

- Modifier la méthode **queryButtonActionPerformed** :

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
if (!NameTextField.getText().trim().equals("")) {
        if(id.isSelected()) {
           rechercheparid();
                  }
        else if (livre.isSelected()) {
           recherchepartitre() ;
          }
        else if (auteur.isSelected()) {
           rechercheparauteur() ;
          }
    }
```

II.2.3.18 Ajouter la méthode displayResult :

```
private void displayResult(List resultList) {
    Vector<String> tableHeaders = new Vector<String>(); // construire l'entête de la
table
    Vector tableData = new Vector(); // récupérer les données ligne par ligne
    tableHeaders.add("Id");
    tableHeaders.add("Titre");
    tableHeaders.add("Auteur");
    for (Object o : resultList) {
       Bibliotheque f = (Bibliotheque) o;
        Vector<Object> oneRow = new Vector<Object>();
        oneRow.add(f.getId());
        oneRow.add(f.getTitre());
        oneRow.add(f.getAuteur());
        tableData.add(oneRow);
    }
    resultTable.setModel(new DefaultTableModel(tableData, tableHeaders));
}
```

II.2.3.19 Ajouter les imports nécessaires :

- import java.util.Vector;

- import javax.swing.table.DefaultTableModel;
- import bibliotheque.entity.Bibliotheque;

II.2.3.20 Exécution de notre application :

II.2.3.20.1 Premier mode d'exécution : « par code »

Figure II.27: Premier mode d'exécution.

II.2.3.20.2 Deuxième mode d'exécution : « par titre »

- Saisir le/les premiers caractères du titre.

Figure II.28: Deuxième mode d'exécution.

II.2.3.20.3 Troisième mode d'exécution : « par auteur »

- Saisir le/les premiers caractères du nom d'auteur

Figure II. 29: Troisième mode d'exécution.

II.3 Conclusion :

Dans ce deuxième chapitre, nous avons présenté la description de l'application. En présentant les étapes nécessaires pour la création de notre projet avec les différents modes d'exécution.

Notre application permet de faire la recherche sur les livres, il existe d'autres fonctionnalités telque l'ajout, modification et emprunté des livres.

CONCLUSION GÉNÉRALE

Conclusion générale :

Le travail présenté dans ce PFE rentre dans le cadre du développement et la gestion de bibliothèque qui regroupe de nombreux concepts.

Dans notre travail, nous avons développé un logiciel pour gérer notre bibliothèque, qui servira aux utilisateurs comme outil de recherche et de traitement de l'information.

Le choix du langage d'implémentation joue un rôle important dans la construction d'une application donnée. La structure que nous avons présentée concerne l'utilisation du langage JAVA, qui permet de développé une application de gestion de bibliothèque.

Références Bibliographiques :

[1] http://www.memoireonline.com/07/10/3641/m_Conception-et-developpement-dun-site-web-pour-hopital-Cas-de-lhpital-general-de-refere0.html. Le 12/5/2014.

[2] http://www.exlibrisgroup.com/fr/files/brochure/Aleph.pdf

[3] http://www.cabinetneveux.com/omegaweb/index_fichiers/infrastructurereseau .htm. Le 11/05/2014

[4] AYAD ZEDDAM Hakima, BENAINI Khalida: «Conception et implémentation d'une application client-serveur *Gestion de la bibliothèque*». PFE.

[5] Eric cariou « Introduction aux systèmes distribués ». PDF.

[6] http://labo-techno.casciani.fr/index_menu54.html. Le 14/05/2014

[7] http://www.awt.be/contenu/tel/res/res,fr,fic,080,000.PDF. Le 29/11/01

[8] http://sebsauvage.net/comprendre/p2p/. Le 11/05/2014

[9] M.AZZI Développement et déploiementd' application client/serveur.
Le 12/5/2014

[10] http://www.babylon.com/definition/Remote_Procedure_Call/French. Le 12/5/2014

[11] Dr A.KHELASSI Les Systèmes distribuées les platformes distribué (intergiciel /midelwares).PDF. Le 06/04/2014

[12] http://www.commentcamarche.net/contents/1030-introduction-a-rmi-remote-method-invocation. Le 12/05/2014

[13] http://www.commentcamarche.net/contents/1029-architecture-de-rmi-remote-method-invocation#gonext. Le 12/05/2014

[14] http://www.grappa.univ-lille3.fr/polys/sql/ch02s05.html. Le 12/05/2014

[15] http://www.commentcamarche.net/contents/1013-le-modele-relationnel.
Le 12/5/2014

[16] http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/internet-java-485/. Le 12/05/2014

[17] http://nadir.is.online.fr/index.php/content/view/93/76/. Le 14/05/2014

[18] http://www.commentcamarche.net/contents/557-java. Le 12/05/2014

[19] http://www.techno-science.net/?onglet=glossaire&definition=5346. Le 05/05/2014

[20] http://telecharger.tomsguide.fr/EasyPHP,0301-839.html. Le 28/02/2014

[21] http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/internet-mysql-4640/. Le 12/05/2014

[22] http://www.jmdoudoux.fr/java/dej/chap-swing.htm. Le 10/06/2014

[23] http://redac.socialinput.fr/le-framework-hibernate-pour-les-amoureux-du-java/. Le 12/05/2014

[24] http://gardeux-vincent.eu/Documents/ProjetJEE/LMNTF_Hibernate_JSF/hibernate.html. Le 12/05/2014

Résumé :

Dans ce PFE, nous avons réalisé l'application « Gestion d'une bibliothèque » avec Swing et le Framework Hibernate. L'objectif principal est d'aider l'utilisateur à faire des recherches au sein de la bibliothèque en utilisant un ensemble de critère comme le titre du livre, son code ou les noms des auteurs.

MOTS CLES : JAVA, NETBEANS, HIBERNATE, SWING, MYSQL

Abstract:

This work aims to "Managing a library" with Swing and Hibernate Framework. The main objective is to help the user to do some research in the library using a set of criteria such as the book title, code or authors' names.

KEYWORDS: JAVA, NETBEANS, HIBERNATE, SWING, MYSQL.

ملخص:

في هذا المشروع أجرينا "إدارة المكتبات"معتمدين على "سوينج"و"هيبرنت"، الذي يوفر وظيفة البحث و العرض، و لتطوير هذا التطبيق استخدمنا لغة جافا مع نتبينس 6.8

من خلال هذا المشروع تمكنا من فهم لغة جافا و تطوير معارفنا.

الكلمات المفتاحية :JAVA, NETBEANTS, HIBERNATE, SWING, MYSQL